2015-12-16

GPU Accelerated Spectral Element Methods: 3D Euler equations

Giraldo, Francis

GPU Accelerated Spectral Element Methods: 3D Euler equations American Geophysical Union (AGU) Fall Meeting, 16 December 2015

http://hdl.handle.net/10945/48821

GPU Accelerated Spectral Element Methods: 3D Euler equations

Daniel Abdi, Lucas Wilcox, Timothy Warburton* & Francis Giraldo

Naval Postgraduate School, Virginia Tech University*

Contact Information:
Department of Applied Mathematics
Naval Postgraduate School
1 University Circle, Monterey, CA, USA

Phone: +1 (831) 582 7010
Email: dsabdi@nps.edu

Non-hydrostatic Unified Model of the Atmosphere

Abstract

A GPU accelerated nodal discontinuous Galerkin method for the solution of the 3D Euler equations in the Non-hydrostatic Unified Model of the Atmosphere (NUMA) is presented. We use algorithms suitable for the single instruction multiple thread architecture of GPUs to accelerate the dynamical core of NUMA by two orders of magnitude relative to one core of a CPU. Tests on one node of the Titan supercomputer yield a speedup of upto **15X** on one K20x GPU relative to that on an AMD Opteron CPU with 16 cores. The scalability of the multi-GPU implementation is tested using 16384 GPUs, which resulted in a weak scaling efficiency of about **90%**. For portability to heterogeneous computing environment, we used a new programming language OCCA, which can be cross-compiled to either **OpenCL, CUDA or OpenMP** at runtime. Finally, the accuracy and performance of our GPU implementations are verified using benchmark problems representative of different scales of atmospheric dynamics.

Introduction

The National Oceanic and Atmospheric Administration (NOAA) has set a goal of 3 km resolution for an operational NWP model within the next decade [4]. The Non-hydrostatic Unified Model of the Atmosphere (NUMA) has recently achieved this goal using 3 million MPI threads of the *multi-core* MIRA supercomputer [3]. The current study aims to port NUMA to *many-core* architecture (CPUs, GPUs, XeonPhi etc..) using a new programming language called OCCA [2] that provides backends to different kinds of hardware. This will enable NUMA to use future heterogeneous hardware using the same code that will be compiled and optimized for each computing unit in the heterogeneous cluster.

NUMA uses element based Galerkin (EBG) methods, continuous Galerkin (CG) and discontinuous Galerkin (DG), for spatial discretization. These methods are well suited for GPU computing for two reasons a) Localized memory accesses result in low communication overhead. In contrast, global methods require an *all-to-all* communication that severely degrades scalability. b) High order polynomial expansion of the solution results in large arithmetic intensity per degree of freedom. Parallel implementation of DG is often easier and more efficient than that of CG because of a smaller communication stencil; only neighbors sharing a face need to communicate in DG as opposed to edge and corner neighbor communication required by CG. Moreover, DG allows for a simple overlap of computation of volume integrals and intra-processor flux with communication of boundary data, which can be exploited to improve the efficiency of the parallel implementation [1].

Governing equations

The Euler equations are

$$\frac{\partial \rho}{\partial t} + \nabla.(\rho \mathbf{u}) = 0$$
$$\frac{\partial \rho \mathbf{u}}{\partial t} + \nabla.(\rho \mathbf{u} \otimes \mathbf{u} + P\mathbf{I}_3) = -\rho g \hat{\mathbf{k}}$$
$$\frac{\partial \rho \theta}{\partial t} + \nabla.(\rho \theta \mathbf{u}) = 0$$

or in compact vector notation form

$$\frac{\partial \mathbf{q}}{\partial t} + \nabla.\mathbf{F} = S(\mathbf{q})$$

with the equation of state given by

$$P = P_0 \left(\frac{\rho R \theta}{P_0}\right)^{\gamma}$$

Discretization

Continuous Galerkin:

$$\int_{\Omega_e} \psi_i \frac{\partial \mathbf{q}_N}{\partial t} d\Omega_e + \int_{\Omega_e} \psi_i \nabla.\mathbf{F} d\Omega_e = \int_{\Omega_e} \psi_i S(\mathbf{q}_N) d\Omega_e$$

Discontinuous Galerkin:

$$\int_{\Omega_e} \psi_i \frac{\partial \mathbf{q}_N}{\partial t} d\Omega_e + \int_{\Gamma_e} \psi_i \hat{\mathbf{n}}.(\mathbf{F}^* - \mathbf{F}) d\Gamma_e + \int_{\Omega_e} \psi_i \nabla.\mathbf{F} d\Omega_e = \int_{\Omega_e} \psi_i S(\mathbf{q}_N) d\Omega_e$$

The major compute kernels are

1. Volume integration kernel
2. Surface integration kernel
3. Time update kernel
4. Direct Stiffness Summation (DSS) required by CG

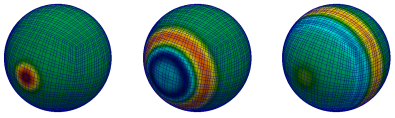
Speedup

Table 1: Speedup comparison between CPU and GPU for single and double precision calculations at different number of elements and polynomial orders using CUDA translation of OCCA kernel code. A maximum speedup of about 15X is observed using the K20X GPU card on the Titan supercomputer relative to a 16-core 2.2GHz AMD Opteron 6274 CPU. The CPU/GPU times (seconds) and speedups are given first for the double precision result and then for single precision.

N	30x30=900 elements			40x40=1600 elements		
	CPU	GPU	Speedup	CPU	GPU	Speedup
2	10.62/9.98	2.17/1.57	4.90/6.36	18.83/17.41	3.70/2.53	5.09/6.88
3	22.01/19.66	3.06/1.87	7.19/10.51	41.53/34.72	5.04/3.06	8.24/11.35
4	46.45/35.19	5.12/3.03	9.07/11.61	81.91/63.55	8.69/5.07	9.43/12.53
5	77.03/61.35	9.88/4.86	7.80/12.62	137.49/107.30	17.11/8.35	8.04/12.85
6	122.27/95.67	16.11/7.71	7.59/12.41	210.35/166.40	28.15/13.49	7.47/12.33
7	195.61/135.21	18.87/9.65	10.37/14.01	343.74/236.09	33.05/16.86	10.40/14.00

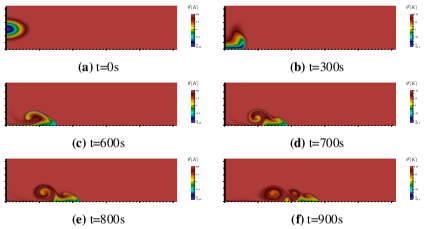
N	80x80=6400 elements			120x120=14400 elements		
	CPU	GPU	Speedup	CPU	GPU	Speedup
2	80.72/70.41	13.33/8.94	6.05/7.88	184.19/172.85	27.82/19.78	6.62/8.74
3	179.07/142.19	18.46/11.18	9.70/12.72	405.15/324.78	41.08/24.87	9.86/13.06
4	350.54/268.69	32.71/19.02	10.71/14.13	798.50/599.25	72.77/42.34	10.97/14.15
5	587.17/429.66	67.03/32.38	8.76/13.27	1329.79/1007.31	150.56/72.08	8.83/13.97
6	925.25/696.25	110.92/52.91	8.34/13.16	2086.84/1586.54	249.50/118.39	8.36/13.40
7	1406.61/968.10	130.16/66.41	10.81/14.58	3158.43/2227.29	293.05/148.76	10.78/14.97

Verification



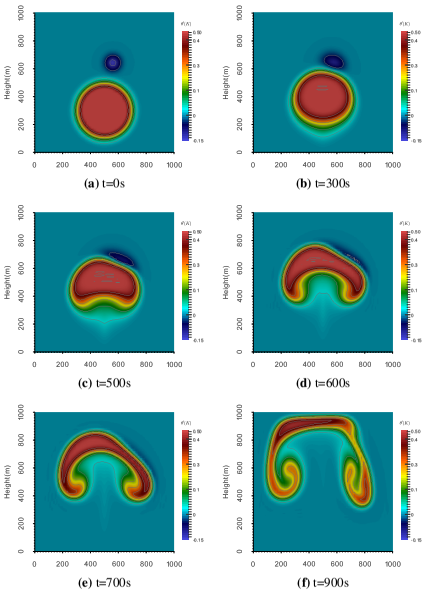
(a) 0h **(b)** 4h **(c)** 7h

Figure 1: Propagation of an acoustic wave. The density perturbation after 0 hour, 4 hours and 7 hours. A cubed sphere grid with 10x10 elements in the horizontal and 3 elements in the vertical with 3rd degree polynomial is used.



(a) t=0s **(b)** t=300s **(c)** t=600s **(d)** t=700s **(e)** t=800s **(f)** t=900s

Figure 2: Density current. Evolution of potential temperature perturbation $\theta'(K)$ run with CG and an artificial viscosity of $\nu = 75\ m^2/s$ for stabilization. Results are shown at t=0,300, 600, 700, 800 and 900 seconds. A grid of 128x1x32 elements with 4^{th} degree polynomials is used for an effective resolution of 50m in x and z directions.



(a) t=0s **(b)** t=300s **(c)** t=500s **(d)** t=600s **(e)** t=700s **(f)** t=900s

Figure 3: Colliding thermal bubbles. Evolution of potential temperature perturbation $\theta'(K)$ run with CG and an artificial viscosity of $\nu = 1.5\ m^2/s$ for stabilization. Results are shown at t=0,300, 500, 600, 700 and 900 seconds. A grid of 10x1x10 elements with 6^{th} degree polynomials is used.

Scalability

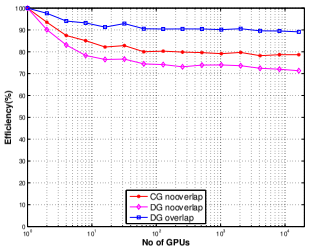


Figure 4: Scalability test of Multi-GPU implementation on Titan: The scalability of NUMA for up to 16384 GPUs on the Titan supercomputer is shown. Each node of Titan contains a Tesla K20X GPU. An efficiency of about 90% is observed relative to a single GPU. The test is conducted using a unified implementation of CG and DG. The efficiency of DG is significantly improved (by about 20%) when overlapping communication with computation, which helps to hide both latency in the data copy between the CPU and GPU and the CPU-CPU communication..

Conclusions

- NUMA is ported to many-core architecture using a new programming language called OCCA that can be cross compiled to CUDA, OpenCL, OpenMP
- A speed up of upto 240X relative to a single core CPU (or 15X relative to a 16-core CPU) is obtained using a single NVIDIA K20X GPU
- A weak scaling efficiency of about 90% is observed using 16384 GPUs of the Titan supercomputer

References

[1] J. F. Kelly and F. X. Giraldo. Continuous and discontinuous galerkin methods for a scalable three-dimensional nonhydrostatic atmospheric model: limited area mode. *J. Comput. Phys.*, 231:7988 – 8008, 2012.

[2] D. Medina, St-Cyr Amik, and T. Warburton. OCCA: A unified approach to multi-threading languages. *arXiv*, 2014.

[3] A. Müller, M. Kopera, S. Marras, L.C. Wilcox, T. Isaac, and F.X Giraldo. Strong scaling for numerical weather prediction at petascale with the atmospheric model numa. *Submitted to : 30th IEEE International Parallel and Distributed Processing Symposium*, 2016.

[4] NOAA. High performance computing plan 2015-2020. *National Oceanic and Atmospheric Adminstration*, pages 1 – 11, 2015.

Acknowledgements

The authors gratefully acknowledge the support of the Office of Naval Research through program element PE-0602435N. This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.